PATENT ABSTRACTS OF JAPAN

(11)Publication number : 07-049779
(43)Date of publication of application : 21.02.1995

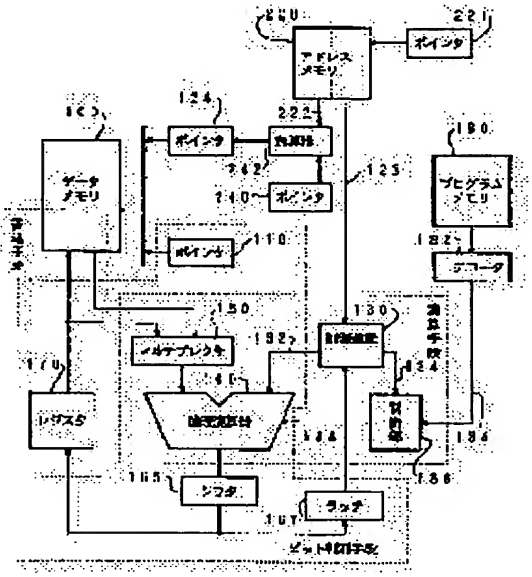
(51)Int.Cl. G06F 9/315
H03M 13/22
H04B 14/04

(21)Application number : 05-211077
(22)Date of filing : 04.08.1993

(71)Applicant : MATSUSHITA ELECTRIC IND CO LTD
(72)Inventor : OKAMOTO MINORU
UEDA KATSUHIKO
ASANO NOBUO

(54) METHOD AND DEVICE FOR INTERLEAVING

(57)Abstract:
PURPOSE: To reduce the storage area of relative addresses furthermore by adding a reference value common to all addresses and a relative address peculiar to each address to generate address information for the storage area of data after interleave processing.
CONSTITUTION: Only relatively changing parts of storage destination addresses in a data memory 100 of data, which are obtained by processing of data bits before interleave processing, and bit positions are stored in an address memory 220 in order from relative address 0. The reference value for write of data after interleave processing to a data memory 100 is stored in a pointer 240. Data is read out from the address designated by a pointer 221 of the address memory 220, and relative address information 222 of this data and the reference value in the pointer 240 are added by an adder 242. and the result is stored in a pointer 124. Thus, the relative value obtained by subtraction of the reference value is taken as information to be held in the address memory 220.



LEGAL STATUS

[Date of request for examination]	09.01.1998
[Date of sending the examiner's decision of rejection]	
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]	
[Date of final disposal for application]	
[Patent number]	2999101
[Date of registration]	05.11.1999
[Number of appeal against examiner's decision of rejection]	
[Date of requesting appeal against examiner's decision of rejection]	
[Date of extinction of right]	

(19)日本国特許庁（ＪＰ）　(12)特　許　公　報（Ｂ２）　(11)特許番号
特許第2999101号
(P2999101)

(45)発行日　平成12年１月17日(2000.1.17)　(24)登録日　平成11年11月５日(1999.11.5)

(51)Int.Cl.[7]	識別記号	ＦＩ	
Ｇ０６Ｆ　9/315		Ｇ０６Ｆ　9/30	３４０Ｄ
Ｈ０３Ｍ　13/27		Ｈ０３Ｍ　13/22	
Ｈ０４Ｂ　14/04		Ｈ０４Ｂ　14/04	Ｆ

請求項の数２(全 13 頁)

(21)出願番号　特願平5－211077

(22)出願日　平成５年８月４日(1993.8.4)

(65)公開番号　特開平7－49779

(43)公開日　平成７年２月21日(1995.2.21)

審査請求日　平成10年１月９日(1998.1.9)

(73)特許権者　000005821
松下電器産業株式会社
大阪府門真市大字門真1006番地

(72)発明者　岡本　稔
大阪府門真市大字門真1006番地　松下電器産業株式会社内

(72)発明者　上田　勝彦
大阪府門真市大字門真1006番地　松下電器産業株式会社内

(72)発明者　浅野　延夫
大阪府門真市大字門真1006番地　松下電器産業株式会社内

(74)代理人　100079544
弁理士　斎藤　勲

審査官　伊知地　和之

最終頁に続く

(54)【発明の名称】　インターリーブ装置

(57)【特許請求の範囲】

【請求項１】インターリーブ処理前のデータの記憶域とインターリーブ処理後のデータの記憶域とを含む第１の記憶手段と、

前記インターリーブ処理後のデータの記憶域の番地情報と、該データのインターリーブ処理後のビットのビット位置情報とを記憶する第２の記憶手段と、

前記インターリーブ処理前のデータのインターリーブを希望するビットの状態を判別するビット判別手段と、

前記番地情報に基づき前記第１の記憶手段から読出されたインターリーブ処理後のデータの、該データに対する前記ビット位置情報により指定された位置のビットを前記ビット判別手段の出力に応じセットもしくはリセットの操作を行なう演算手段と、

前記演算手段の出力を前記インターリーブ処理後のデータの記憶域に書込む書込手段とを備え、

前記第２の記憶手段に記憶した番地情報とビット位置情報はポインタを順々に１増分することにより得られることを特徴とするインターリーブ装置。

【請求項２】前記第２の記憶手段に記憶の番地情報は相対的番地情報であり、

前記インターリーブ装置は、更に、

前記相対的番地情報と共同して前記番地情報を構成する基準値を含むポインタと、

前記相対的番地情報と前記基準値とを加算して前記インターリーブ処理後のデータの記憶域の前記番地情報を出力する加算器と、を含むことを特徴とする請求項１記載のインターリーブ装置。

【発明の詳細な説明】

【０００１】

3

【産業上の利用分野】本発明は送信データのインターリーブ処理を行なうインターリーブ装置に関するものである。

【０００２】

【従来の技術】近年、ディジタル信号処理プロセッサ（以下ＤＳＰと略称する）を通信分野に適用する場合、送信データに対する処理の１つとして、インターリーブ処理を行なうようにしている。このインターリーブ処理は、順序道りに並べられたビット列をある一定の規則に基づいて並べかえる処理である。

【０００３】ＤＳＰにおけるインターリーブ処理は、例えば、以下のような手順によって行なわれる。それは、インターリーブするべき順序道りに並べられたデータをメモリから読出し、そのデータを特定のビット数だけシフトし、シフトされたデータとインターリーブ処理をして格納する特定番地（例えば、Ａ番地）のメモリデータとの論理和を計算し、その計算結果をメモリのＡ番地に格納して行われる。この手順を送信データのビットの数だけ繰返すことによってインターリーブ処理が完成する。

【０００４】以下、上記の従来技術によるインターリーブ装置について、図３に従いその構成を説明する。図３は従来のインターリーブ装置の構成を示すブロック図である。

【０００５】図３において、３００は１語又は１ワード当たりＮｄビット（Ｎｄは正の整数、例えば１６）で構成される複数の語からなるインターリーブ処理前の又は処理するべきデータの記憶域と、インターリーブ処理後のデータ又は処理されるべきデータ（既にインターリーブしたビットを有し、更に継続してインターリーブするべきものを含む）の記憶域とを含むデータメモリ、３２４はデータメモリ３００のアクセス番地を指定し、インターリーブ処理後のデータが格納される番地を保持するポインタ、３４０はデータメモリ３００から読出されたインターリーブ処理後のデータを入力し、それを後述のレジスタ３４２で示すビット数だけシフトするシフタである。

【０００６】更に、３４２はシフタ３４０がシフトするビットの数を供給するレジスタ、３６０は後述のレジスタ３７０から出力したデータとデータメモリ３００から読出されシフタ３４０でシフトされたデータとを入力して論理演算する論理演算器、３７０はＮｄビット（Ｎｄは正の整数、例えば１６）で構成され、論理演算器３６０またはデータメモリ３００からデータを入力するレジスタ、３８０はここで実行するプログラムを格納するプログラムメモリ、３８２はプログラムメモリ３８０から読出された命令を解読するデコーダ、３８４は論理演算器３６０の動作を制御する制御信号、である。

【０００７】以下、図３、図４、図６、図９に従い、上記のように構成されたインターリーブ装置によってインターリーブ処理を実行するその動作について説明する。

4

ここで、図４は従来のインターリーブ処理によりインターリーブするべき又はインターリーブ処理前のデータの格納状態を示す構成図である。図６はインターリーブ処理によりインターリーブ処理後のデータの格納状態を示す構成図である。図９はプログラムメモリに格納された従来のプログラムの例を示す図である。尚、図３については前述した。

【０００８】以下で説明するインターリーブ処理は、例えば、図４に示した状態でデータメモリ３００に格納されているインターリーブ処理前のデータを図６に示すように並べかえるものとし、図４に示すように、インターリーブ処理前のデータＤ１乃至ＤＮｔは０番地から順にその最下位ビットに格納されており、また、図６に示すデータメモリ３００の領域には初期値として０が既に格納され、データメモリ３００及びレジスタ３７０は１６ビット（Ｎｄ＝１６）構成という条件の下で行われるものと仮定する。

【０００９】一方、プログラムは、図９に示すように、プログラムメモリ３８０の各番地（ｎ～ｎ＋ｍ）に格納されており、その処理は番地順に従って行うものとする。以下、その処理手順について番地順に説明する。

【００１０】［プログラムメモリ３８０：ｎ番地］データメモリ３００のポインタ３２４（ポインタ３２４には予め１００を格納しておく）で指定された番地から（予め０・・・・・０に初期化されている）データを読出しレジスタ３７０に格納する。

【００１１】［プログラムメモリ３８０：ｎ＋１番地］レジスタ３４２にシフタ３４０でシフトするビット数を設定する。本例では、最初、図４に示すように、データメモリ３００の０番地に格納されているデータＤ１をレジスタ３７０のビット１５に格納するため、シフトビット数１５を設定する。

【００１２】［プログラムメモリ３８０：ｎ＋２番地］データメモリ３００の０番地のデータ‘０・・・・・０Ｄ1’をシフタ３４０に読出し、そこでレジスタ３４２に保持されているシフトビット数‘１５’だけ上位側にシフトし、それを論理演算器３６０に送る。そこで、１５ビット位置にシフトされたデータ‘Ｄ1０・・・・・０’とレジスタからのデータ（予め格納された‘０・・・・・０’）との論理和（オア）を計算し、その結果のデータ‘Ｄ1０・・・・・０’をレジスタ３７０に格納する。これにより、レジスタ３７０のビット１５にＤ１を格納することができた（他のビット位置は全部０）。

【００１３】それ以降のインターリーブ処理も上記同様にプログラムを進め、ビットを付加してインターリーブ処理を実行することができる。すなわち、プログラムメモリ３８０における次のプログラムｎ＋３番地及びｎ＋４番地においても、ｎ＋１番地及びｎ＋２番地のプログラムと類似する動作を行う。しかし、ここでは、データ

5

メモリ３００の１６番地に格納されているビットＤ１７をデータメモリ３００の１００番地のＤ１の次のビット位置（ビット位置１４）に格納するインターリーブ処理を実行することになる。

【００１４】実際には、前回の演算結果のデータ‘D10・・・・・０’はレジスタ３７０に格納されているので、それを取出して論理演算器３６０の一方の入力に挿入する。他方、データメモリ３００の１６番地に格納されているデータビットＤ１７（ビット位置０に格納されている）を読出し、シフタ３４０において上位に１４ビット位置シフトして論理演算器３６０の他方の入力に挿入する。論理演算器３６０から両入力データの論理和‘D1 D17 0・・・・・０’が出力され、それをレジスタ３７０に格納し、前に格納されていたデータ‘D10・・・・・０’と置き換える。このインターリーブ処理において、ビットＤ１７が０の場合は‘D10・・・・・０’、ビットＤ１７が１の場合は‘D11・・・・・０’となる。Ｄ１の値も勿論０か１である。

【００１５】以上説明したｎ＋３番地及びｎ＋４番地のプログラム処理と同様な動作をＮｄ回（本例では１６回）実行して、最後にｎ＋３１番地及びｎ＋３２番地のプログラム処理を終了すると、レジスタ３７０の全ビットが完成する。

【００１６】［プログラムメモリ３８０：ｎ＋ｍ番地］レジスタ３７０に格納されている完成した１６ビットのデータをデータメモリ３００のポインタ３２４で指定した番地（１００番地）に格納し、ポインタ３２４の値を１増分す（ポインタ３２４の増分機構は図示せず）。これにより、データメモリ３００のポインタ３２４で指定した番地（１００番地）にはインターリーブ処理が完成した後のデータ（１６ビット）を格納することができたことになる。

【００１７】以後、プログラムメモリのｎ番地からｎ＋ｍ番地と同様な処理をデータメモリ３００の１０１番地以下の番地についても同様に実行することによりインターリーブ処理が完了する。（例えば、「ＭＮ１９０１／ＭＮ１９０９ユーザーズマニュアル６、９９、１０４頁（松下電子工業刊）」に示された命令セット、及びハードウェア構成により上記処理を実行することができる）。

【００１８】

【発明が解決しようとする課題】しかしながら、上記のような構成でインターリーブ処理を行なうと、プログラムメモリ３８０及び、データメモリ３００に非常に大きな記憶領域を必要とする。すなわち、プログラムメモリ３８０においては、ｎ＋１番地及びｎ＋２番地の２命令中で使用するレジスタ３４２に対するシフトビット数（上記の例では、最初１５、次に１４）、およびデータメモリ３００から読みだすべき番地の指定（上記の例では、最初０番地、次に１６番地）は、インターリーブすべきデータ毎に固有の値であるため、プログラムごとに行うべき番地指定をループ処理（繰り返し処理）で実行することができない。従って、プログラムをループ処理できないので、プログラムメモリ３８０の使用量は１回のインターリーブ処理（Ｎｔ個のデータに対し）において、ｎ＋１番地及びｎ＋２番地の命令に相当する命令をインターリーブすべきデータの個数（Ｎｔ個）分実行しなければならないことになる。そのため、プログラムは２×Ｎｔ語（１語は下記のように、本例ではＮｉビット）だけ記述する必要がある。（本例では、インターリーブ処理後のデータのデータメモリ３００における最初の格納番地を１００番地としたが、Ｎｔ＝２３０とした場合はそれを２３０番地以降としなければならないことになる。）

【００１９】又、図９に示すｎ番地及びｎ＋ｍ番地に相当する２つの命令はＮｄ回ごとに一回実行するので、Ｎｔ／Ｎｄ個記述する必要がある。従って、プログラムメモリ３８０の１語当りのビット数をＮｉ（Ｎｉは正の整数）ビットとすると、使用メモリは全体として（２×Ｎｔ＋２×Ｎｔ／Ｎｄ）×Ｎｉビットの領域をインターリーブ処理のために確保しなければならない。また、データメモリ３００においては、Ｎｔ×Ｎｄビットの領域（図４）をインターリーブ処理前のデータ格納領域として確保しなければならない。従って、全体として、例えば、Ｎｔ＝２３０，Ｎｄ＝１６，Ｎｉ＝３２とすると、プログラムメモリ３８０において１５６４０ビット、データメモリ３００においては３６８０ビット、全体で１９３２０ビット必要となる。

【００２０】このように、上記のような構成でインターリーブ処理を実行する場合、プログラムメモリ３８０及びデータメモリ３００の回路規模が非常に増大するという問題があった。

【００２１】本発明は、上記の問題に鑑みてなされたもので、インターリーブ処理の実行に必要なプログラム及びデータを記憶するプログラムメモリ及びデータメモリが占有する記憶領域を大幅に縮小したインターリーブ装置を提供することを目的とする。

【００２２】

【課題を解決するための手段】本発明によるインターリーブ装置は、上記の問題を解決するため、インターリーブ処理前のデータの記憶域とインターリーブ処理後のデータの記憶域とを含む第１の記憶手段と、インターリーブ処理後のデータの記憶域の番地情報と、該データのインターリーブ処理後のビットのビット位置情報とを記憶する第２の記憶手段と、インターリーブ処理前のデータのインターリーブを希望するビットの状態を判別するビット判別手段と、番地情報に基づき第１の記憶手段から読出されたインターリーブ処理後のデータの、該データに対するビット位置情報により指定された位置のビットをビット判別手段の出力に応じセットもしくはリセット

の操作を行なう演算手段と、演算手段の出力をインターリーブ処理後のデータの記憶域に書込む書込手段と、を備えたことを特徴とする

【００２３】本発明によるインターリーブ装置は、更に、上記の問題を解決するため、第２の記憶手段に記憶されている番地情報を相対的番地情報とし、インターリーブ装置は、更に、相対的番地情報と共同して番地情報を構成する基準値を含むポインタと、第２の記憶手段からの相対的番地情報とポインタからの基準値とを加算してインターリーブ処理後のデータの記憶域の番地を指定する番地情報を出力する加算器と、を含むことを特徴とする。

【００２４】

【００２５】

【００２６】

【作用】本発明によれば、インターリーブ処理後のデータの記憶域に対する番地情報とそのインターリーブ処理後のビットの位置を指定するビット位置情報とを記憶する記憶手段を設け、その番地情報とビット位置情報とによりインターリーブ処理後のデータの番地とビット位置とを個々に任意指定することができるようにしたことにより、インターリーブ処理前のデータの全ビット位置にインターリーブ処理前のビットを記憶することができると共に、該インターリーブ処理前のデータを同一プログラムの反復で番地指定可能にして、インターリーブに使用するメモリ領域を大幅に削減することができる。

【００２７】また、インターリーブ処理後のデータの記憶域に対する番地情報を全番地に共通の基準値と各番地に固有な相対的番地とを加算して生成するようにしたことにより、相対的番地の記憶域を更に削減することができる。

【００２８】

【実施例】以下、添付図面を参照して、本発明の実施例を詳細に説明する。先ず、図１、５、６、７、１０に基づき、本発明の第一実施例によるインターリーブ装置について詳細に説明する。図１は、本発明の第１実施例によるインターリーブ装置の構成を示す構成図である。

【００２９】図１において、１００は１語あたりＮｄビット（Ｎｄは正の整数、例えば１６）構成で、複数語からなるデータメモリ、１１０はデータメモリ１００からインターリーブ処理前のデータを読出すために指定する番地を保持するポインタ、１２０はデータメモリ１００に対しインターリーブ処理後のデータを書込む際に指定する番地データ及びビット位置情報を保持するアドレスメモリ、１２１はアドレスメモリ１２０のアクセス番地を指定するポインタ、１２２はアドレスメモリ１２０から出力する番地データ、１２３はアドレスメモリ１２０から出力するビット位置情報、１２４は番地データ１２２を保持し、データメモリ１００に対しインターリーブ処理後のデータの格納番地を与えるポインタである。

【００３０】１３０はビット位置情報１２３と後述のラッチ１６７に記憶されたフラグとを入力する制御装置であって、後述のフラグ（ラッチ１６７の値）が０の場合は、ビット位置情報１２３で示したビットのみが値０で、他のビットすべてを値１にした後述のデータ１３２（例えば、ビット位置情報１２３で指定したビット位置が１５であると‘０１・・・・・１’となる）を後述の論理演算器１６０の一方の入力に出力すると同時に、フラグが０の場合には論理積（アンド）を計算させるよう論理演算器１６０の動作を制御する制御信号１３４を制御部１８６に出力する。

【００３１】又、制御装置１３０は後述のフラグ（ラッチ１６７の値）が１の場合は、ビット位置情報１２３で示したビットのみが値１で、他のビットすべてを値０にした後述のデータ１３２（例えば、前述のように、指定したビット位置が１５であると‘１０・・・・・０’となる）を後述の論理演算器１６０の一方の入力に出力すると同時に、フラグが０の場合には、論理和（オア）を計算させるよう論理演算器１６０の動作を制御する制御信号１３４を制御部１８６に対して出力する。

【００３２】更に、１３２は制御装置１３０から論理演算器１６０に出力するデータ（前述）、１３４は制御装置１３０から制御部１８６に出力して論理演算器１６０に対し論理積を実行するか又は論理和を実行するかの指示を与える制御信号１８８を制御部１８６から出力させる制御信号、１５０は後述のレジスタ１７０からのデータ及びデータメモリ１００の出力データを入力し、どちらか一方を選択して論理演算器１６０の他方の入力に出力するマルチプレクサ、１６０はデータ１３２とマルチプレクサ１５０からの出力データとを入力し、制御信号１８８に応答して論理積か又は論理和を計算する論理演算器である。

【００３３】更に、１６５は論理演算器１６０の出力データを下位側に１ビットシフトし、後述のレジスタ１７０にシフトした結果を出力し、及びその最下位ビットをラッチ１６７に出力するシフタ、１６７はシフタ１６５の出力データの最下位ビット（フラグ）を保持するラッチ、１７０はｒ０及びｒ１の２語からなり、１語あたりＮｄビット（Ｎｄは正の整数、例えば１６）で構成され、シフタ１６５からの出力データを入力してそれをデータメモリ１００及びマルチプレクサ１５０に出力するレジスタである。

【００３４】１８０はインターリーブ処理用プログラム（例えば、図１０，１１に示す）を格納するプログラムメモリ、１８２はプログラムメモリ１８０から読出された命令を解読するデコーダ、１８４はデコーダ１８２から制御部１８６に出力される制御信号、１８６は制御信号１８４及び制御信号１３４を入力し、論理演算器１６０の動作を制御する制御部、１８８は制御信号１３４の入力に応答して制御部１８６から論理演算器１６０に出

力してその計算（論理積又は論理和）を制御する制御信号である。

【００３５】又、図１において、ポインタ１１０、マルチプレクサ１５０、論理演算器１６０、シフタ１６５、レジスタ１７０、及びラッチ１６７によってビット判別手段を構成し、制御装置１３０、データ１３２、制御信号１３４、マルチプレクサ１５０、論理演算器１６０、制御部１８６、及び制御信号１８８によって演算手段を構成し、レジスタ１７０、及びデータメモリ１００によって書込手段を構成している。

【００３６】以下、上記のように構成された本第１実施例によるインターリーブ装置において、インターリーブ処理を実行する動作を図１のほか、図５乃至図７、図１０を用いて説明する。図５は本発明によるインターリーブ処理前のデータの格納状態を示す構成図である。図６はインターリーブ処理後のデータの格納状態を示す構成図である。図７は本第１実施例のインターリーブ処理に使用するアドレスメモリのデータ格納状態を示す構成図である。図１０はプログラムメモリに格納された本第１実施例のインターリーブ処理に使用するプログラムの例を示す図である。尚、図１については前述した。

【００３７】以下で示すインターリーブ処理動作は、例えば、データメモリ１００に図５に示した状態で格納されているインターリーブ処理前のデータを図６に示すように並べかえるものとし、インターリーブ処理前のデータＤ１乃至ＤＮｔは、図５に示すように、０番地を先頭番地として最下位ビットから最上位ビットまで順に格納されており、データメモリ１００及びレジスタ１７０は１６ビット（Ｎｄ＝１６）で構成されるという条件の下で行われるものと仮定する。

【００３８】アドレスメモリ１２０には、インターリーブ処理前のデータ・ビットＤ１～ＤＮｔそれぞれに対するインターリーブ処理後のデータのデータメモリ１００に対する格納先番地及びそのビット位置が０番地から順に、本実施例では図７に示すような形式で格納されている。また、ポインタ１２１には予め０が格納されているものとする。

【００３９】プログラムメモリ１８０には、図１０に示すプログラムが格納されている。以下、そのプログラムメモリ１８０の番地に従い、ｎ番地から順に処理の手順を説明する。

【００４０】［プログラムメモリ１８０：ｎ番地］ポインタ１１０によって指定されたデータメモリ１００の番地（０番地）のデータをマルチプレクサ１５０、論理演算器１６０、及びシフタ１６５を経由してレジスタ１７０のｒ１に格納し、同時に最下位ビット（Ｄ１）をラッチ１６７に格納する。更に、ポインタ１２１で指定したアドレスメモリ１２０の番地（最初は０番地）のデータを読み出し、番地データ１２２をポインタ１２４に格納する。本例による番地データによると、初回はアドレス

メモリ１２０の０番地を指定したからポインタ１２４の値は１００となる（図７）。それと同時に、ポインタ１１０の値を１増分する（ポインタ１１０の増分機構は図に示していない）。

【００４１】［プログラムメモリ１８０：ｎ＋１番地］ｎ＋２～ｎ＋４番地に示すプログラムの処理を繰り返す回数は、本例では処理を１語ごとに行うから、‘Ｎｄ－１＝１５’を設定する。なお、処理の繰り返し実行の機構は図に示していない。

【００４２】［プログラムメモリ１８０：ｎ＋２番地］インターリーブ格納命令を実行する。この命令はデータメモリ１００のポインタ１２４で指定した番地（本例では１００番地）のデータの指定ビット位置にラッチ１６７の値（本例では、下記のように、Ｄ１＝０であれば０、Ｄ１＝１であれば１）をセットして、レジスタ１７０のｒ０に格納し、同時にポインタ１２１を１増分する（１０１になる）というものである。すなわち、プログラムメモリ１８０からこの命令が読みだされると、デコーダ１８２で解読され、インターリーブ格納命令の実行を制御する制御信号１８４が出力される。

【００４３】一方、制御装置１３０は、以下に述べるようなデータ１３２及び制御信号１３４を出力する。すなわち、ラッチ１６７の値（Ｄ１）が０のときは、ビット位置情報１２３で指定したビット位置（本例では図７に示すように、その０番地でビット位置１５を指定）のビットを０とし、他のビットすべてを１に設定したデータ１３２を出力する。それは、ｘ‘７ＦＦＦ’で示される。（ｘは１６進数であることを示し、７は０１１１、Ｆは１１１１を示す）。同時に、論理演算器１６０において論理積を計算させるよう制御する制御信号１３４を制御部１８６に出力する。

【００４４】同様に、ラッチ１６７の値（Ｄ１）が１のときは、ビット位置情報１２３で指定したビット位置（上記と同じく１５）のビットを１とし、他のビットすべてを０に設定したデータ１３２を出力する。従って、この場合は、ｘ‘８０００’となる。（上記同様、８は１０００、０は００００を示す）。同時に、論理演算器１６０に対し論理和を計算させるよう制御する制御信号１３４を制御部１８６に出力する。

【００４５】制御部１８６は、プログラムメモリ１８０からの制御信号１８４の制御に基づき、上記のように制御信号１３４に応答して、論理演算器１６０に対し論理積または論理和を計算させるよう制御する制御信号１８８を出力する。

【００４６】論理演算器１６０は、ポインタ１２４で指定されたデータメモリ１００の１００番地（本例では、最初、１００番地から始まる）のデータ（最初は内容不問又は不明‘ｘｘ・・・・ｘ’）とデータ１３２（０１１・・・１、又は１００・・・０）を入力し、制御信号１８８で制御される計算（論理積又は論理和）を実行し

て、その結果をｒ０に格納する。

【００４７】すなわち、本例では、Ｄ１が０の場合は、ポインタ１２４で指定されたデータメモリ１００の番地（１００番地）のデータ‘ｘｘ・・・・ｘ’とデータ１３２の内容ｘ‘７ＦＦＦ’（０１・・・・１）との論理積を計算し、Ｄ１が１の場合は、ポインタ１２４で指定されたデータメモリ１００の番地（１００番地）のデータ‘ｘｘ・・・・ｘ’とデータ１３２の内容ｘ‘８０００’（１０・・・・０）との論理和を計算する。これにより、Ｄ１が所定のビット位置に格納できたことになる。ここで、ポインタ１２１を１増分してアドレスメモリ１２０の１番地を指定し次の処理に備える（ポインタ１２１の増分機構は図示していない）。

【００４８】［プログラムメモリ１８０：ｎ＋３番地］ポインタ１２４で指定されたデータメモリ１００の番地（１００番地）にレジスタ１７０のｒ０に格納されているデータを格納する。これにより、データメモリ１００の０番地に格納されているデータのビット位置０（図５）の１ビット（Ｄ１）に対するインターリーブ処理が完了したことになる。

【００４９】［プログラムメモリ１８０：ｎ＋４番地］レジスタ１７０のｒ１に格納されているデータ（本例では、‘Ｄ１６、・・・、Ｄ３、Ｄ２、Ｄ１’）をマルチプレクサ１５０及び論理演算器１６０を介してシフタ１６５に格納し、そこで下位側に１ビットのみシフトして、次のビットＤ２を最右端に配置する。その結果を再びｒ１に格納すると同時に、最下位ビット（Ｄ２）をラッチ１６７にフラグとして格納する。

【００５０】そこで、前述同様、ポインタ１２１で指定したアドレスメモリ１２０の番地（上記で１番地に増分した）のデータ（図７の１番地に示すように、番地データ＝１０２、ビット位置データ＝１）を読出し、番地データ１２２をポインタ１２４に格納する。従って、ポインタ１２４の値は、現在では１０２となる。続いて、繰返し処理の先頭番地（ｎ＋２番地）に戻る。以降、ｎ＋２番地からｎ＋４番地をＮｄ－１（＝１５）回繰り返すことによって、Ｄ１～ＤＮｄ（Ｄ１６）のデータに対するインターリーブ処理を実行することができる。その終了後、プログラムはプログラムメモリ１８０のｎ＋５に進む。

【００５１】［プログラムメモリ１８０：ｎ＋５番地］ｎ番地に分岐し、そこで次にインターリーブするべきデータが格納されているデータメモリ１００の番地を指定して、上記同様の回数の処理を繰返す（本例では、前回、既に１番地増分してあり、１番地を指定する）。

【００５２】次に、インターリーブするべきビットＤ１７を上記でインターリーブしたデータ‘０ｘｘ・・・ｘ’又は‘１ｘｘ・・・ｘ’（ｘは不問又は不明ビット）（図６の１００番地にＤ１のみがインターリーブされている状態）に対し如何に加えるかについて説明する。上記のように、プログラムメモリ１８０のｎ番地に戻り、そこで、ポインタ１１０によりデータメモリ１００の１番地を指定すると、図５の１番地に示すようなデータ‘・・・・、Ｄ１８、Ｄ１７’が読出され、レジスタ１７０のｒ１に格納されると同時に、その最下位ビットＤ１７（０又は１）がラッチ１６７にフラグとして格納される。

【００５３】次に、プログラムメモリ１８０をｎ＋２番地へ進め、ポインタ１２１の指定によるアドレスメモリ１２０の、例えば、１６番地を指定して、その番地情報（本例では、番地データ（１００）、ビット位置データ（１４））を読出し、ポインタ１２４を介してデータメモリ１００の１００番地のデータ（前回インターリーブした‘Ｄ１ｘｘ・・・・ｘ’が格納されている）を読出し、論理演算器１６０において、前回同様、ビットＤ１７が０の場合は‘１０１１・・・・１’と論理積され、ビットＤ１７が１の場合は‘０１００・・・・０’と論理和され、その結果がレジスタ１７０のｒ０に格納される。このようにして、レジスタ１７０のｒ０に格納されたデータは、ビットＤ１７が０の場合は‘Ｄ１　０ｘｘ・・・ｘ’、ビットＤ１７が１の場合は‘Ｄ１　１ｘｘ・・・ｘ’となる（このＤ１の値は、勿論０か１である）。以下、前述同様にプログラムを進め、ビットを付加してインターリーブを実行することができる。

【００５４】以上の手順により、インターリーブ処理前のデータＤ１～ＤＮｔに対し如何にインターリーブ処理を実行するかについて説明した。次に、その際必要とする各種メモリの記憶領域の大きさについて説明する。すなわち、プログラムメモリ１８０については上記６ステップの繰り返しとなるため、６×Ｎｉビット（Ｎｉはプログラムメモリ１８０の１語のビット数、Ｎｉは正の整数）、データメモリ１００についてはＮｔビット、アドレスメモリ１２０についてはＮｔ×Ｎａビット（Ｎａはアドレスメモリ１２０の１語のビット数、Ｎａは正の整数）となる。

【００５５】例えば、前述の従来技術の場合同様、Ｎｔ＝２３０，Ｎｄ＝１６，Ｎｉ＝３２，Ｎａ＝１２とすると、プログラムメモリ１８０はＮｉ×６＝３２×６＝１９２ビット、データメモリ１００はＮｔ＝２３０ビット、アドレスメモリ１２０はＮｔ×Ｎａ＝３２０×１２＝２７６０ビット必要となる。従って、全メモリでは３１８２ビット必要となる。このビット数からわかるように、本発明は前述の従来技術で必要とした１９３２０ビットより相当節約することができたことは明らかとなった。

【００５６】次に、本発明の第２実施例によるインターリーブ装置につき図２、５、６、８、１１を参照して説明する。図２は本発明の第２実施例によるインターリーブ装置の構成を示す構成図、図８は本第２実施例のインターリーブ処理に使用するアドレスメモリのデータ格納

状態を示す構成図、図１１はプログラムメモリに格納された本第２実施例のインターリーブ処理に使用するプログラムの例を示す図である。尚、図５及び図６については、前述の第１実施例のものと同一である。

【００５７】図２において、１００～１８８は図１に示したものと同じものである。２２０はデータメモリ１００にインターリーブ処理後のデータを書込む際に使用する相対的番地情報（格納先番地の相対的に変化する部分、すなわち、後述のポインタ２４０からの増分値）、及びビット位置情報１２３を保持するアドレスメモリ、２２１はアドレスメモリ２２０にアクセス番地を与えるポインタ、２２２はアドレスメモリ２２０から出力する相対的番地情報、２４０はデータメモリ１００にインターリーブ処理後のデータを書込む各番地に共通の部分からなる基準値又は基準番地を保持するポインタ、２４２は相対的番地情報２２２とポインタ２４０からの基準値とを加算し、データメモリ１００にインターリーブ処理後のデータを書込む際に使用する番地情報をポインタ１２４に出力する加算器である。

【００５８】以下、上記のように構成された本実施例のインターリーブ装置において、インターリーブを実行する場合の動作について説明する。以下で示すインターリーブ処理動作は、例えば、データメモリ１００に図５に示した状態で格納されているインターリーブ処理前のデータを図６に示すように並べかえるものとし、図５に示すように、インターリーブ処理前のデータＤ１乃至ＤＮｔは０番地を先頭番地として最下位ビットから最上位ビットまで順に格納されており、又、データメモリ１００及びレジスタ１７０は１６ビット（Ｎｄ＝１６）で構成されるという条件の下で行われるものと仮定する。

【００５９】アドレスメモリ２２０には、インターリーブ処理前のデータ・ビットＤ１～ＤＮｔそれぞれに対するインターリーブ処理後のデータのデータメモリ１００に対する格納先である番地の相対的に変化する部分のみ（以下、相対的番地と呼ぶ）とそのビット位置とが相対的０番地から順に格納されている。本例では、図８に示すように、番地データ（０、２、１、　・・・）及びそれに対応するビット位置データ（１５、１、７、　・・・）がそこに示すような形式で格納されている。また、ポインタ２２１には、予め０が格納されており、ポインタ２４０には、データメモリ１００にインターリーブ処理後のデータを書込む際の基準となる基準値（例えば、最初の番地、本例では１００番地）が格納されている。

【００６０】プログラムメモリ１８０には、図１１に示すプログラムが格納されており、その動作が図１０に示したものと異なるのは以下の点である。すなわち、図１０に示す第１実施例においては、ｎ番地及びｎ＋４番地において、ポインタ１２１で指定されたアドレスメモリ１２０のその番地からデータを読出し、番地データ１２２をポインタ１２４に格納するようにしたが、本実施例

では、それに代わる動作として、アドレスメモリ２２０のポインタ２２１で指定した番地からそのデータを読出し、その中の相対的番地情報２２２とポインタ２４０の基準値（本例では１００）とを加算器２４２において加算し、その結果をポインタ１２４に格納するようにした点が異なる。

【００６１】これにより、アドレスメモリ２２０で保持するべき番地情報をポインタ２４０からの基準値を差引いた相対的な値とすることができ、アドレスメモリ２２０の記憶領域をさらに小さくすることができる。

【００６２】尚、第１及び第２の実施例において、インターリーブ処理前のデータを最下位ビットから順に並べたが、これは最上位ビットからでもよい。このとき、シフタ１６５は上位に１ビットシフトすることになり、ラッチ１６７のフラグはシフタ１６５の出力の最上位ビットを保持することになる。また、インターリーブ処理後のデータ格納の配列は任意でよいことはいうまでもない。

【００６３】

【発明の効果】本発明は、以上説明したように構成し、特に、インターリーブ処理後のデータの記憶域に対する番地データとそのデータのインターリーブ処理後のビットの位置を指定するビット位置情報とを記憶する記憶手段を設け、その番地データとビット位置情報とによりインターリーブ処理後のデータの番地とビット位置とを個々に任意指定することができるようにしたことにより、インターリーブ処理前のデータの全域にインターリーブ処理前のビットを記憶すると共に、該インターリーブ処理前のデータを同一プログラムの反復で番地指定可能にしたことによって、小さい記憶領域（小回路規模）のプログラムメモリ、及び小さいデータメモリの使用でよいため、インターリーブに使用するべきメモリ領域を大幅に削減することができる。更に、本発明は、以上説明したように構成し、特に、インターリーブ処理後のデータの記憶域に対する番地データを全番地に共通の基準値と各番地に固有な各番地の変化分（相対的番地）とを加算して生成するよう構成し、相対的番地のみをアドレスメモリに記憶するようにしたことにより、更に相対的番地の記憶域を削減することができる。

【図面の簡単な説明】

【図１】本発明の第１実施例によるインターリーブ装置の構成図

【図２】本発明の第２実施例によるインターリーブ装置の構成図

【図３】従来のインターリーブ装置の構成図

【図４】従来技術で使用するインターリーブ処理前のデータの格納状態を示す図

【図５】本発明で使用可能なインターリーブ処理前のデータの格納状態を示す図

【図６】インターリーブ処理後のデータの格納状態を示

15

す図

【図７】本発明の第１実施例で使用するアドレスメモリのデータの格納状態を示す図

【図８】本発明の第２実施例で使用するアドレスメモリのデータの格納状態を示す図

【図９】従来のプログラムの例を示す図

【図１０】本発明の第１実施例で使用するプログラムの例を示す図

【図１１】本発明の第２実施例で使用するプログラムの例を示す図

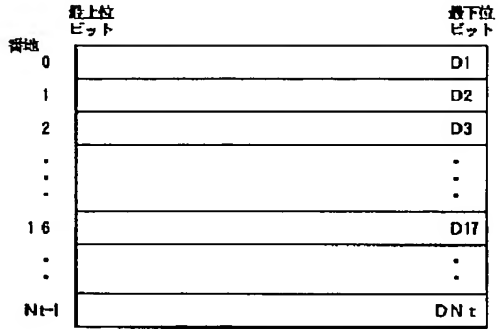
【符号の説明】

１００　データメモリ
１１０，１２１，１２４　ポインタ
１２０　アドレスメモリ
１７０　レジスタ
１２２　番地データ
１２３　ビット位置情報
１３０　制御装置
１３２　データ
１３４，１８４，１８８　制御信号

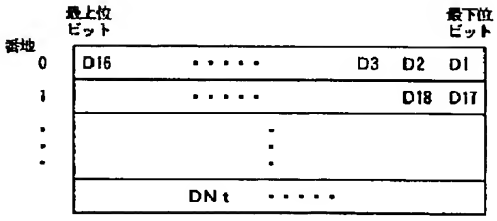
16

１５０　マルチプレクサ
１６０　論理演算器
１６５　シフタ
１６７　ラッチ
１８０　プログラムメモリ
１８２　デコーダ
１８６　制御部
２２０　アドレスメモリ
２２１　ポインタ
２２２　相対的番地情報
２４０　ポインタ
２４２　加算器
３００　データメモリ
３２４　ポインタ
３４０　シフタ
３４２　レジスタ
３６０　論理演算器
３７０　レジスタ
３８０　プログラムメモリ
３８２　デコーダ

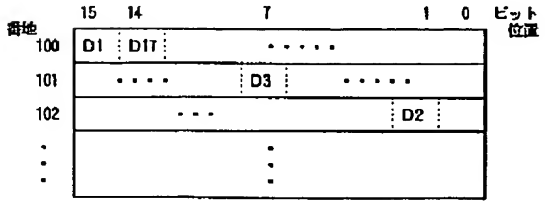
【図４】



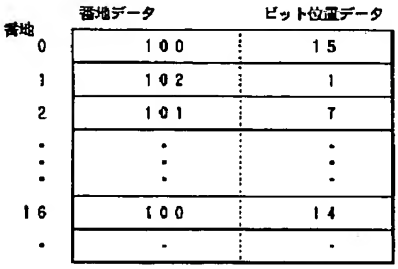
【図５】



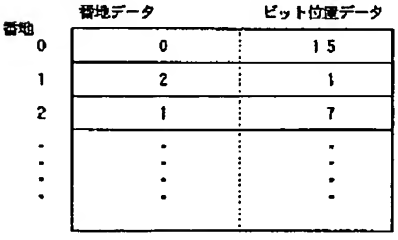
【図６】



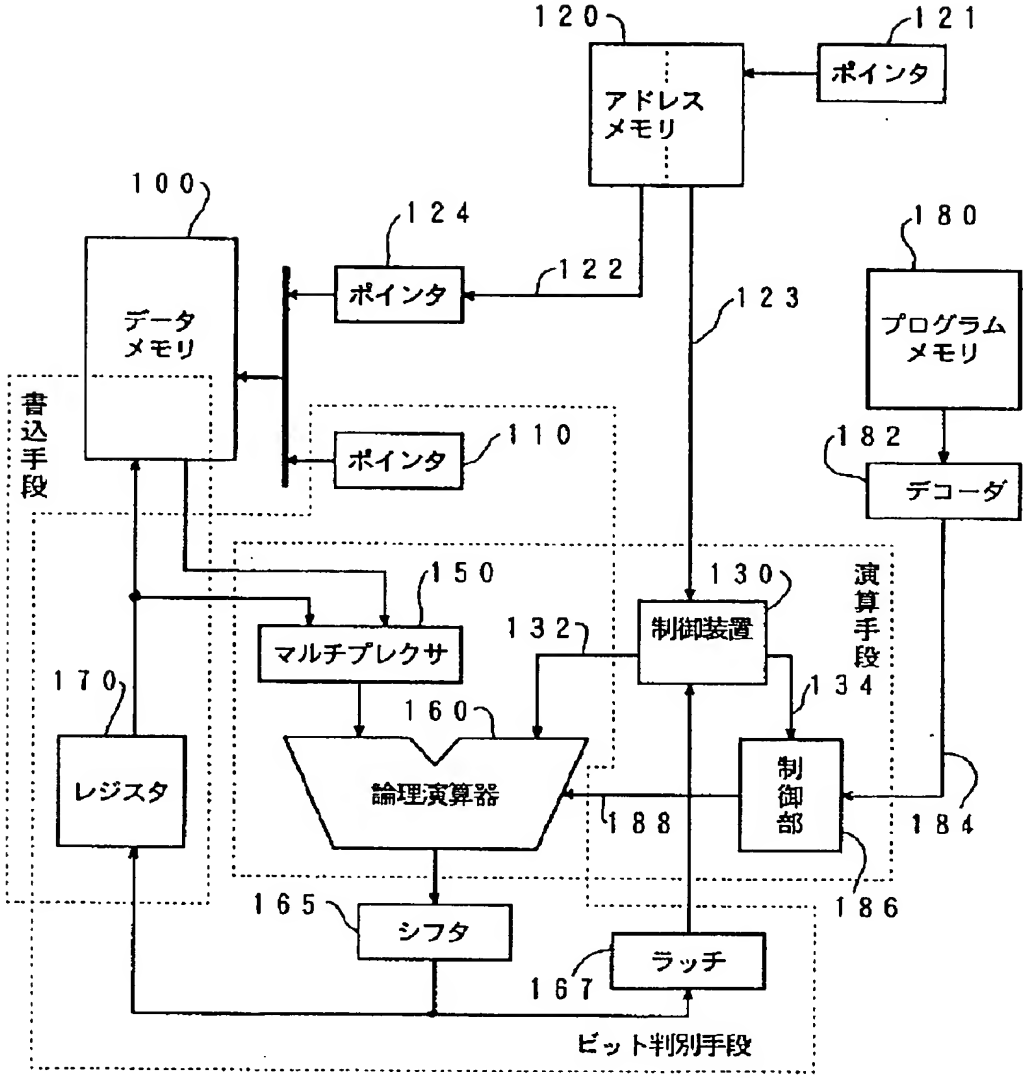
【図７】



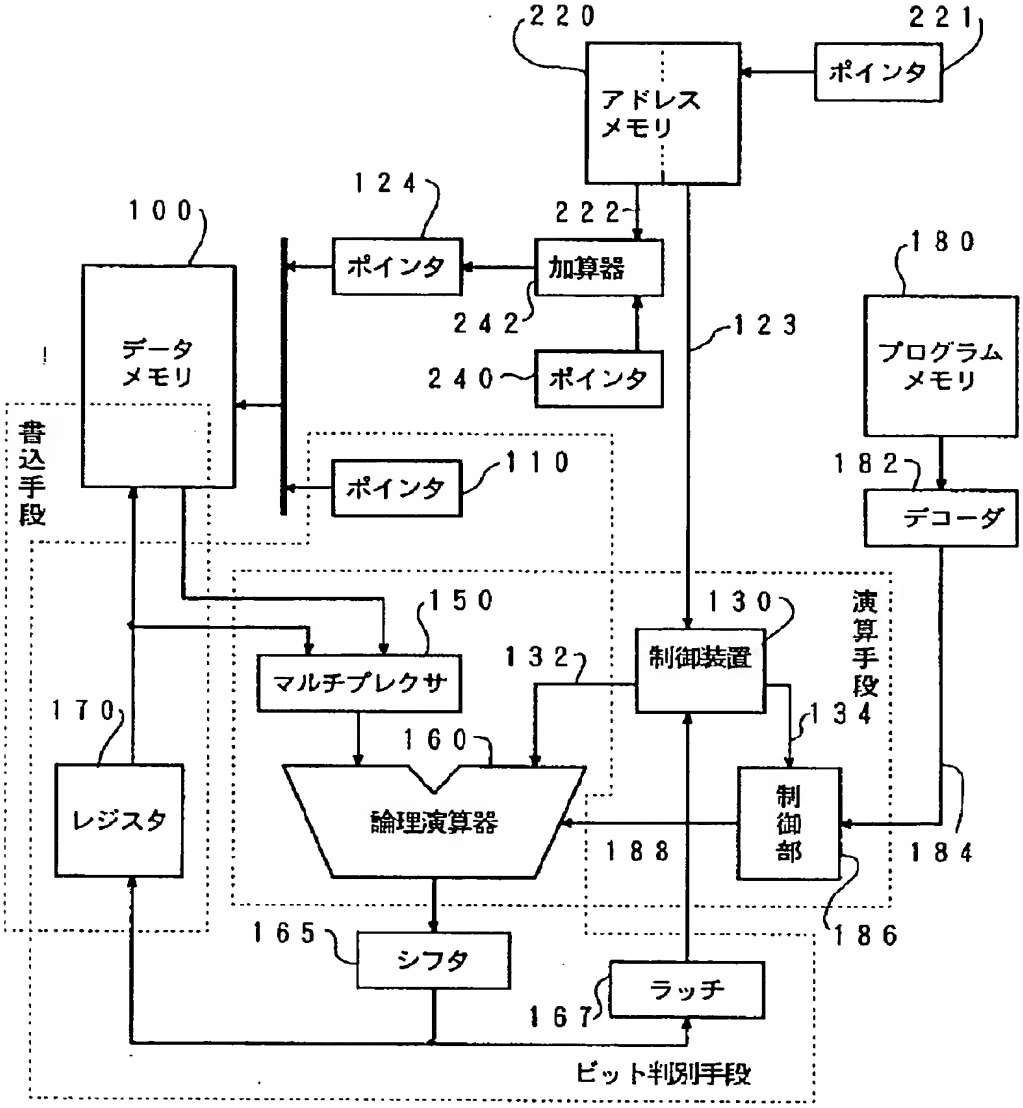
【図８】



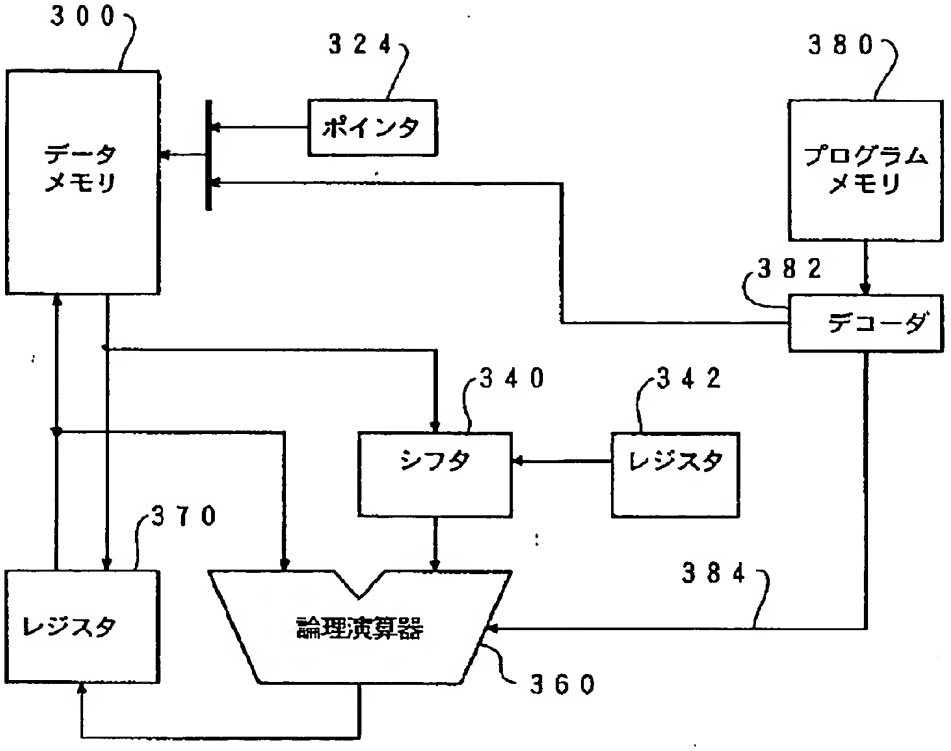
【図１】


１２０
アドレスメモリ
１２１
ポインタ
１００
データメモリ
１２４
ポインタ
１２２
１２３
１８０
プログラムメモリ
書込手段
ポインタ
１１０
１８２
デコーダ
１５０
マルチプレクサ
１３０
制御装置
演算手段
１３２
１７０
１６０
１３４
レジスタ
論理演算器
制御部
１８８
１８４
１８６
１６５
シフタ
ラッチ
１６７
ビット判別手段

【図２】


２２０
２２１
アドレス
メモリ
ポインタ
１２４
１００
２２２
ポインタ
加算器
１８０
２４２
１２３
データ
メモリ
２４０
ポインタ
プログラム
メモリ
書込手段
ポインタ
１１０
１８２
デコーダ
１５０
１３０
演算手段
マルチプレクサ
１３２
制御装置
１７０
１３４
１６０
レジスタ
論理演算器
制
御
部
１８８
１８４
１８６
１６５
シフタ
ラッチ
１６７
ビット判別手段

【図３】



【図９】

番地	処理手順
n	レジスタ３７０ ← データメモリ３００（ポインタ３２４の番地）
n＋1	レジスタ３４２ ← シフトビット数
n＋2	レジスタ３７０ ← レジスタ３７０と （データメモリ３００（０番地）をシフト）の論理和
n＋3	レジスタ３４２ ← シフトビット数
n＋4	レジスタ３７０ ← レジスタ３７０と （データメモリ３００（１７番地）をシフト）の論理和
・ ・ ・ ・	・ ・ ・ ・
n＋m	データメモリ３００（ポインタ３２４の番地） ← レジスタ３７０ ポインタ３２４ ← ポインタ３２４＋１

【図１０】

番地	処理手順
n	ｒ１ ←― データメモリ１００（ポインタ１１０の番地） ポインタ１２４ ←― 番地データ１２２ ポインタ１１０ ←― ポインタ１１０＋１
n＋１	繰返し回数（Ｎｄ－１）の設定
n＋２	インターリーブ格納命令 ・ｒ０ ←― データメモリ１００（ポインタ１２４の番地）の特定ビットのラッチ１６７にフラグをセット ・レジスタ１２１ ←― レジスタ１２１＋１
n＋３	データメモリ１００（ポインタ１２４の番地） ←ｒ０
n＋４	ｒ１ ←― ｒ１を下位側に１ビットシフト ポインタ１２４ ←― 番地データ１２２ ｎ＋２番地に戻る
n＋５	ｎ番地に戻る

【図１１】

番地	処理手順
n	ｒ１ ←― データメモリ１００（ポインタ１１０の番地） ポインタ１２４ ←― 相対的番地データ２２２＋ポインタ２４０ ポインタ１１０ ←― ポインタ１１０＋１
n＋１	繰返し回数（Ｎｄ－１）の設定
n＋２	インターリーブ格納命令 ・ｒ０ ←― データメモリ１００（ポインタ１２４の番地）の特定ビットのラッチ１６７にフラグをセット ・レジスタ１２１ ←― レジスタ１２１＋１
n＋３	データメモリ１００（ポインタ１２４の番地） ←ｒ０
n＋４	ｒ１ ←― ｒ１を下位側に１ビットシフト ポインタ１２４ ←― 相対的番地データ２２２＋ポインタ２４０ ｎ＋２番地に戻る
n＋５	ｎ番地に戻る

フロントページの続き

(56)参考文献　特開　昭61－282933（ＪＰ，Ａ）
特開　平２－253339（ＪＰ，Ａ）
特開　昭56－107655（ＪＰ，Ａ）
特開　平４－279934（ＪＰ，Ａ）
特開　昭61－276043（ＪＰ，Ａ）
特開　昭61－86850（ＪＰ，Ａ）
特開　平２－306339（ＪＰ，Ａ）
特開　平１－217528（ＪＰ，Ａ）
特開　昭61－9725（ＪＰ，Ａ）
特開　平５－6303（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.[7]，ＤＢ名)
G06F　9/30 - 9/355
G06F　9/40 - 9/42

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

CLAIMS

[Claim(s)]
[Claim 1] Interleave equipment characterized by providing the following The 1st storage means including a data storage region before interleave processing, and a data storage region after interleave processing Address information on a data storage region after said interleave processing The 2nd storage means which memorizes bit-position information on a bit after interleave processing of this data A bit distinction means to distinguish a condition of a bit of wishing an interleave of data before said interleave processing, Data after interleave processing read from said 1st storage means based on said address information, An operation means to perform actuation of a set or reset for a bit of a location specified using said bit-position information over this data according to an output of said bit distinction means, and a write-in means which writes an output of said operation means in a data storage region after said interleave processing
[Claim 2] It is interleave equipment according to claim 1 characterized by for address information on storage for said 2nd storage means to be relative address information, and for said interleave equipment to contain the adder which adds a pointer including a reference value which constitutes said address information in collaboration with said relative address information further, and said relative address information and said reference value, and outputs said address information on a data storage region after said interleave processing.
[Claim 3] An interleave method characterized by providing the following A data storage region before interleave processing and a data storage region after interleave processing are prepared, and it is the address information on a data storage region after said interleave processing. Bit-position information on a bit after interleave processing of this data is set up beforehand. Take out a bit which wishes an interleave of data before said interleave processing one by one, and the condition is distinguished. Data after interleave processing read based on said address information, A bit of a location specified using said bit-position information over this data is set as a value corresponding to a condition of said distinguished bit. Each production process which memorizes again data set as a value corresponding to the condition that a bit of said specified location is said distinguished bit to a storage area specified for said address information
[Claim 4] Said address information is the interleave method according to claim 3 characterized by consisting of a reference value which constitutes said address information in collaboration with relative address information and this relative address information.

[Translation done.]

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Industrial Application] This invention relates to the interleave equipment which performs interleave processing of transmit data.
[0002]
[Description of the Prior Art] When applying a digital-signal-processing processor (it calls for short Following DSP) to the communication link field in recent years, it is made to perform interleave processing as one of the processings to transmit data. This interleave processing is processing which rearranges the bit string put in order by sequence **** based on a certain fixed regulation.
[0003] Interleave processing in DSP is performed by the following procedures, for example. It calculates an OR with the memory data of the specific address (for example, Ath street) which considers interleave processing as the data which only the specific number of bits shifted the data put in order by sequence **** which should be interleaved, and was shifted [data] in read-out from memory, and its data, and is stored, stores the count result in the Ath street of memory, and is performed. Interleave processing is completed when only the number of the bits of transmit data repeats this procedure.
[0004] Hereafter, according to drawing 3 , the configuration is explained about the interleave equipment by the above-mentioned conventional technology. Drawing 3 is the block diagram showing the configuration of conventional interleave equipment.
[0005] In drawing 3 , 300 before the interleave processing which consists of one word or two or more words which consist of Nd bits (Nd is a positive integer, 16 [for example,]) per word Or the data storage region which should be processed, Data or data which should be processed (it has the already interleaved bit) after interleave processing Furthermore, data memory including the storage area containing what should be continued and interleaved, The pointer holding the address with which 324 specifies the access address of data memory 300, and the data after interleave processing is stored, 340 is a shifter which shifts only the number of bits which inputs the data after the interleave processing read from data memory 300, and shows it with the below-mentioned register 342.
[0006] Furthermore, the register which supplies the number which is the bit to which a shifter 340 shifts 342, The logical operation machine which inputs and carries out logical operation of the data which 360 was read from the data and data memory 300 which were outputted from the below-mentioned register 370, and was shifted by the shifter 340, The register which 370 consists of Nd bits (Nd is a positive integer, 16 [for example,]), and inputs data from the logical operation machine 360 or data memory 300, the program memory which stores the program which performs 380 here, the decoder which decodes the instruction with which 382 was read from program memory 380, and the control signal with which 384 controls actuation of the logical operation machine 360 — it comes out.
[0007] The actuation which performs interleave processing with the interleave equipment constituted as mentioned above hereafter according to drawing 3 , drawing 4 , drawing 6 , and drawing 9 is explained. Here, drawing 4 is the block diagram showing the storing condition of the data before the **** interleave processing which should be interleaved by the conventional interleave processing. Drawing 6 is the block diagram showing the storing condition of the data after interleave processing by interleave processing. Drawing 9 is drawing showing the example of the conventional program stored in program memory. In addition, it mentioned above about drawing 3 .
[0008] As the interleave processing explained below shall rearrange the data before the interleave processing stored in data memory 300 in the condition which showed in drawing 4 as shown in drawing 6 , and it shows it to drawing 4 The data D1 before interleave processing thru/or DNt are stored in the least significant bit sequentially from the 0th street. Moreover, 0 is already stored in the field of data memory 300 shown in drawing 6 as initial value, and data memory 300 and a register 370 are assumed to be what is performed under the conditions of a 16-bit (Nd=16) configuration.
[0009] On the other hand, as a program is shown in drawing 9 , it shall be stored in each address (n-n+m) of program memory 380, and the processing shall be performed according to the order of an address. Hereafter, the procedure is explained in order of an address.
[0010] Data is stored in the read-out register 370 from the address specified with the pointer 324 (100 is beforehand stored in the pointer 324) of the [380:n-th program memory] data memory 300 (beforehand initialized by 0.....0).
[0011] The number of bits shifted to the [380:n+1st program memory] register 342 by the shifter 340 is set up. In this example, since the data D1 stored in the 0th street of data memory 300 is stored in the bit 15 of a register 370 as shown in drawing 4 at first, the shift number of bits 15 is set up.
[0012] The 0th data'0 of the [380:n+2nd program memory] data memory 300 ..... Only the shift number of bits '15' currently held [read-out and there] at the register 342 at the shifter 340 shifts 0D1' to a high order side, and it is sent to the logical operation machine 360. Then, the OR (Orr) of the data 'D 10.....0' and the data ('0.....0' stored beforehand) from a register which were shifted to 15 bit positions is calculated, and the data 'D 10.....0' of the result is stored in a register 370. Thereby, D1 was storable in the bit 15 of a register 370 (other bit positions are all 0).
[0013] The interleave processing after it as well as the above can advance a program, can add a bit, and can perform interleave processing. That is, also in the n+3rd programs of the degree in program memory 380, and the n+4th street, the n+1st programs [n+2nd] and similar actuation are performed. However, interleave processing which stores in the 100th next bit position of D1 of data memory 300 (bit position 14) the bit D17 stored in the 16th street of data memory 300 will be performed here.
[0014] In fact, since the data 'D 10.....0' of the last result of an operation is stored in the register 370, it takes it out and inserts it in one input of the logical operation machine 360. On the other hand, in read-out and a shifter 340, the 14 bit positions of data bits D17 (stored in the bit position 0) stored in the 16th street of data memory 300 are shifted to a high order, and it inserts in the input of another side of the logical operation machine 360. OR'D1 D17 0 of the logical operation machine 360 to both input data ..... It replaces with the data 'D 10.....0' which 0' was outputted, stored it in the register 370, and was stored before. In this interleave processing, when a bit D17 is 0 and 'D 10.....0' and a bit D17 are 1, it is set to 'D 11.....0'. Of course, the value of D1 is also 0 or 1.

[0015] After carrying out Nd time (this example 16 times) activation of the same actuation as the n+3rd program manipulation [n+4th] explained above and ending the n+31st program manipulation [n+32nd] finally, all the bits of a register 370 are completed.
[0016] It stores in the address (100th street) which specified the completed 16-bit data which is stored in the [program memory 380:n+m address] register 370 with the pointer 324 of data memory 300, and is 1 ****** (not shown [the increment device of a pointer 324]) about the value of a pointer 324. By this, data (16 bits) after interleave processing is completed is able to be stored in the address (100th street) specified with the pointer 324 of data memory 300.
[0017] Henceforth, interleave processing is completed from the n-th street of program memory by performing the same processing as a n+m address similarly less than about the 101st address of data memory 300. (For example, the above-mentioned processing can be performed by the instruction set shown in "6 or 99,104 pages (**********) of MN1901/MN1909 user's manuals", and the hardware configuration) .
[0018]
[Problem(s) to be Solved by the Invention] However, when interleave processing is performed with the above configurations, a very big storage region is needed for program memory 380 and data memory 300. Namely, it sets to program memory 380. The shift number of bits to the register 342 which is n+1st the n+2nd 2 ordering, and is used (in the above-mentioned example) Assignment of the address which should be read from 14 and data memory 300 to 15 and a degree at first (in the above-mentioned example) At first, to the 0th degree, for every data which should be interleaved, since the 16th street is the value of a proper, it cannot perform addressing which should be performed for every program by loop processing (repeat processing). Therefore, since the loop processing of the program cannot be carried out, the amount of the program memory 380 used must perform the data which should interleave the instruction equivalent to the n+1st instructions [n+2nd] by the number (Nt individual) in one interleave processing (as opposed to the data of Nt individual). Therefore, a program needs to describe only 2xNt language (one word is nickel bit at this example as follows). (In this example, although the first storing address in the data memory 300 of the data after interleave processing was made into the 100th street, when referred to as Nt=230, it must be considered as 230th henceforth.)
[0019] Moreover, since two instructions equivalent to the n-th n+m address shown in drawing 9 are executed once for every Nd time, it is necessary to carry out Nt/Nd individual description. Therefore, if the number of bits per word of program memory 380 is made into nickel (nickel is positive integer) bit, use memory must be secured as a whole (2xNt+2xNt/Nd) for interleave processing of the field of xnickel bit. Moreover, in data memory 300, the field (drawing 4) of a NtxNd bit must be secured as a data storage field before interleave processing. Therefore, as a whole, when Nt=230, Nd=16, and nickel=32, in 15640 bits and data memory 300, 19320 bits is needed [by 3680 bits and the whole] in program memory 380.
[0020] Thus, when interleave processing was performed with the above configurations, there was a problem that program memory 380 and the circuit scale of data memory 300 increased very much.
[0021] This invention was made in view of the above-mentioned problem, and aims at offering the interleave equipment which reduced sharply the storage region which the program memory and data memory which memorize the program and data of interleave processing required for activation occupy.
[0022]
[Means for Solving the Problem] The 1st storage means which includes a data storage region before interleave processing, and a data storage region after interleave processing in order that interleave equipment by this invention may solve the above-mentioned problem, The 2nd storage means which memorizes address information on a data storage region after interleave processing, and bit-position information on a bit after interleave processing of this data, A bit distinction means to distinguish a condition of a bit of wishing an interleave of data before interleave processing, Data after interleave processing read from the 1st storage means based on address information, An operation means to perform actuation of a set or reset for a bit of a location specified using bit-position information over this data according to an output of a bit distinction means, [0023] characterized by having a write-in means which writes an output of an operation means in a data storage region after interleave processing In order that interleave equipment by this invention may solve the further above-mentioned problem, address information memorized by the 2nd storage means is made into relative address information. Interleave equipment Furthermore, a pointer including a reference value which constitutes address information in collaboration with relative address information, It is characterized by including an adder which outputs address information which adds relative address information from the 2nd storage means, and a reference value from a pointer, and specifies an address of a data storage region after interleave processing.
[0024] In order that an interleave method by this invention may solve the above-mentioned problem, a data storage region before interleave processing and a data storage region after interleave processing are prepared. Address information on a data storage region after interleave processing, Bit-position information on a bit after interleave processing of this data is set up beforehand. Take out a bit which wishes an interleave of data before interleave processing one by one, and the condition is distinguished. It is set as a value corresponding to a bit condition that a bit of a location specified using bit-position information over this data of data after interleave processing read based on address information was distinguished. It is characterized by including each production process which memorizes again data set as a value corresponding to the condition that a bit of a specified location was distinguished and of being a bit to a storage area specified for address information.
[0025] An interleave method by this invention is characterized by constituting address information with a reference value which constitutes address information in collaboration with relative address information and this relative address information in order to solve the further above-mentioned problem.
[0026]
[Function] The interleave equipment and the interleave method by this invention A storage means to memorize the bit-position information which specifies the location of the bit after interleave processing of the address information over the data storage region after interleave processing and its data is established. By having enabled it to carry out arbitration assignment of the address and the bit position of data after interleave processing separately using the address information and bit-position information While the bit before interleave processing is memorizable to all the bit positions of the data before interleave processing, addressing of the data before this interleave processing can be made possible repeatedly [of the same program], and the memory areas used for an interleave can be reduced sharply.
[0027] The interleave equipment and the interleave method by this invention can reduce the storage areas of a relative address further by adding a reference value common to all addresses, and a relative address peculiar to each address, and having generated the address information over the data storage region after interleave processing.
[0028]
[Example] Hereafter, the example of this invention is explained to details with reference to an accompanying drawing. First, based on drawing 1 , and 5, 6, 7 and 10, the interleave equipment by the first example of this invention is explained to details. Drawing 1 is the block diagram showing the configuration of the interleave equipment by the 1st example of this invention.
[0029] In drawing 1 , 100 is Nd bit (Nd is positive integer, 16 [for example,]) configuration per word. The pointer holding the

address specified in order that the data memory which consists of two or more words, and 110 may read the data before interleave processing from data memory 100, The address memory holding the address data and bit-position information which are specified in case 120 writes in the data after interleave processing to data memory 100, The pointer with which 121 specifies the access address of the address memory 120, The address data which outputs 122 from the address memory 120, the bit-position information which outputs 123 from the address memory 120, and 124 are pointers which hold the address data 122 and give the storing address of the data after interleave processing to data memory 100.

[0030] When 130 is a control unit which inputs the flag memorized by the bit-position information 123 and the below-mentioned latch 167 and the below-mentioned flag (latch's 167 value) is 0 The below-mentioned data 132 (for example, '01 that the bit position specified for the bit-position information 123 is 15) with which only the bit shown using the bit-position information 123 made all other bits the value 1 with the value 0 ..... 1' — becoming — when a flag is 0, the control signal 134 which controls actuation of the logical operation machine 160 to make an AND (and) calculate is outputted to a control section 186, at the same time it outputs to one input of the below-mentioned logical operation machine 160.

[0031] Moreover, a control unit 130 is the below-mentioned data 132 (for example, '10 that the specified bit position is 15 as mentioned above) with which only the bit shown using the bit-position information 123 is a value 1, and made all other bits the value 0 when the below-mentioned flag (latch's 167 value) was 1. ..... 0' — becoming — when a flag is 0 at the same time it outputs to one input of the below-mentioned logical operation machine 160, the control signal 134 which controls actuation of the logical operation machine 160 to make an OR (Orr) calculate is outputted to a control section 186.

[0032] Furthermore, the data which outputs 132 to the logical operation machine 160 from a control unit 130 (above-mentioned), The control signal to which the control signal 188 which gives directions of whether 134 is outputted to a control section 186 from a control unit 130, and performs an AND to the logical operation machine 160 or to perform an OR is made to output from a control section 186, 150 inputs the data from the below-mentioned register 170, and the output data of data memory 100. the multiplexer which chooses either and is outputted to the input of another side of the logical operation machine 160, and 160 — data 132 and the output data from a multiplexer 150 — inputting — a control signal 188 — answering — an AND — or it is the logical operation machine which calculates an OR.

[0033] Furthermore, 165 carries out 1 bit shift of the output data of the logical operation machine 160 to a low order side. The shifter which outputs the result shifted to the below-mentioned register 170, and outputs the least significant bit to latch 167, The latch for whom 167 holds the least significant bit (flag) of the output data of a shifter 165, 170 is a register which consists of two words, r0 and r1, consists of Nd bits (Nd is a positive integer, 16 [for example,]) per word, inputs the output data from a shifter 165, and outputs it to data memory 100 and a multiplexer 150.

[0034] The program memory in which 180 stores the program for interleave processing (for example, shown in drawing 10 and 11), The decoder which decodes the instruction with which 182 was read from program memory 180, The control signal with which 184 is outputted to a control section 186 from a decoder 182, and 186 input a control signal 184 and a control signal 134. They are the control section which controls actuation of the logical operation machine 160, and the control signal which 188 answers the input of a control signal 134, outputs it to the logical operation machine 160 from a control section 186, and controls the count (an AND or OR).

[0035] Moreover, in drawing 1 , a pointer 110, a multiplexer 150, the logical operation machine 160, a shifter 165, a register 170, and latch 167 constitute a bit distinction means, a control unit 130, data 132, a control signal 134, a multiplexer 150, the logical operation machine 160, a control section 186, and a control signal 188 constitute an operation means, and a register 170 and data memory 100 constitute the write-in means.

[0036] Hereafter, in the interleave equipment by **** 1 example constituted as mentioned above, the actuation which performs interleave processing is explained using others, drawing 5 or drawing 7 , and drawing 10 . [drawing 1] Drawing 5 is the block diagram showing the storing condition of the data before the interleave processing by this invention. Drawing 6 is the block diagram showing the storing condition of the data after interleave processing. Drawing 7 is the block diagram showing the data storage condition of the address memory used for interleave processing of **** 1 example. Drawing 10 is drawing showing the example of the program used for interleave processing of **** 1 example stored in program memory. In addition, it mentioned above about drawing 1 .

[0037] The interleave processing actuation shown below shall rearrange the data before the interleave processing stored in data memory 100 in the condition which showed in drawing 5 , as shown in drawing 6 . The data D1 before interleave processing thru/or DNt As shown in drawing 5 , it is stored in order from the least significant bit to the most significant bit by making the 0th street into a head address, and data memory 100 and a register 170 are assumed to be what is performed under the conditions of consisting of 16 bits (Nd=16).

[0038] The storing first-move ground to the data memory 100 of the data after the interleave processing to the data bit D1 before interleave processing - each DNt and its bit position are stored in the address memory 120 in format as shown in drawing 7 sequentially from the 0th street at this example. Moreover, 0 shall be beforehand stored in a pointer 121.

[0039] The program shown in drawing 10 is stored in program memory 180. Hereafter, according to the address of the program memory 180, the procedure of processing is explained to order from the n-th street.

[0040] The data of the address (0th street) of the data memory 100 specified by the [180:n-th program memory] pointer 110 is stored in 170 registerr1 via a multiplexer 150, the logical operation machine 160, and a shifter 165, and the least significant bit (D1) is stored in latch 167 at coincidence. Furthermore, the data of the address (it is the 0th street at first) of the address memory 120 specified with the pointer 121 is read, and the address data 122 is stored in a pointer 124. According to the address data twisted to this example, since the first time specified the 0th street of the address memory 120, the value of a pointer 124 is set to 100 (drawing 7). 1 increment of the value of a pointer 110 is carried out to it and coincidence (the increment device of a pointer 110 is not shown in drawing).

[0041] The [180:n+1st program memory] In this example, since the count which repeats processing of the program shown in the n+2 to n+4th street processes for every word, it sets up 'Nd-1=15'. In addition, the device of repeat activation of processing is not shown in drawing.

[0042] The [180:n+2nd program memory] interleave store instruction is executed. Latch's 167 value (if it is D 1= 0 and is 0 and D 1= 1 as follows in this example 1) is set to the appointed bit position of the data of an address (this example 100th street) specified with the pointer 124 of data memory 100, and this instruction stores in 170 registerr0, and carries out 1 increment of the pointer 121 to coincidence (set to 101). That is, if this instruction is read from program memory 180, it will decode by the decoder 182 and the control signal 184 which controls activation of an interleave store instruction will be outputted.

[0043] On the other hand, a control unit 130 outputs data 132 and a control signal 134 which are described below. That is, when latch's 167 value (D1) is 0, the bit of the bit position (by this example, as shown in drawing 7 , the bit position 15 is specified by the 0th street) specified for the bit-position information 123 is set to 0, and the data 132 which set all other bits as 1 is outputted. It is shown by x '7FFF'. (It is shown that x is a hexadecimal, 7 shows 0111 and F shows 1111) . The control signal 134 controlled

to make coincidence calculate an AND in the logical operation machine 160 is outputted to a control section 186.
[0044] Similarly, when latch's 167 value (D1) is 1, the bit of the bit position (it is 15 as well as the above) specified for the bit-position information 123 is set to 1, and the data 132 which set all other bits as 0 is outputted. Therefore, it is set to x '8000' in this case. (Like the above, 8 shows 1000 and 0 shows 0000) . The control signal 134 controlled to make coincidence calculate an OR to the logical operation machine 160 is outputted to a control section 186.
[0045] Based on control of the control signal 184 from program memory 180, a control section 186 answers a control signal 134 as mentioned above, and outputs the control signal 188 controlled to make an AND or an OR calculate to the logical operation machine 160.
[0046] The logical operation machine 160 inputs the 100th data (in this example, it begins from the 100th street at first) (they are the contents unquestioned or unknown'xx....x' at first) and data 132 (011...1 or 100...0) of data memory 100 which were specified with the pointer 124, performs count (an AND or OR) controlled by the control signal 188, and stores the result in r0.
[0047] Namely, in this example, when D1 is 0, an AND with the contents x '7FFF' (01....1) of the data 'xx....x' of the address (100th street) of data memory 100 and data 132 specified with the pointer 124 is calculated. When D1 is 1, an OR with the data 'the contents x'8000 of xx....x' and data 132' (10....0) of the address (100th street) of the data memory 100 specified with the pointer 124 is calculated. By this, D1 is able to store in the predetermined bit position. Here, 1 increment of the pointer 121 is carried out, the 1st street of the address memory 120 is specified, and it prepares for the next processing (not shown [the increment device of a pointer 121]).
[0048] The data stored in 170 registerr0 is stored in the address (100th street) of the data memory 100 specified with the [180:n+3rd program memory] pointer 124. It means that the interleave processing to 1 bit (D1) of the bit position 0 (drawing 5) of the data stored in the 0th street of data memory 100 was completed by this.
[0049] The data (this example D16, ..., 'D3, D2, D1') stored in the 170 [180:n+4th program memory] registerr1 is stored in a shifter 165 through a multiplexer 150 and the logical operation machine 160, 1 bit is shifted to a low order side there, and the following bit D2 is arranged to the right end. While the result is again stored in r1, the least significant bit (D2) is stored in latch 167 as a flag.
[0050] Then, the data (as shown in the 1st street of drawing 7 , it is address data =102 and bit-position data =1) of the address (increment was carried out to the 1st street by the above) of the address memory 120 specified with the pointer 121 is stored in read-out like the above-mentioned, and the address data 122 is stored in a pointer 124. Therefore, the value of a pointer 124 is set to 102 by current. Then, it returns to the head address (the n+2nd street) of repetitive operation. henceforth, the n+2nd street to the n+4th street — Nd- interleave processing to the data of D1-DNd (D16) can be performed by things 1 (= 15) **********. A program progresses to n+5 of program memory 180 after the termination.
[0051] The [180:n+5th program memory] It branches to the n-th street, the address of data memory 100 with which the data which should be interleaved next there is stored is specified, and processing of the same count as the above is repeated (in this example, last time, 1st increment has already been carried out and the 1st street is specified).
[0052] Next, it explains how the bit D17 which should be interleaved is added to the data '0 xx...x' or '1 xx...x' (x is unquestioned or the unknown bit) (condition that only D1 is interleaved by the 100th street of drawing 6) interleaved above. As mentioned above, when the 1st street of data memory 100 is specified with a pointer 110, while data (...., and 'D18, D17') as shown in the 1st street of drawing 5 is read by the n-th street of program memory 180 and it is stored in 170 registerr1 return and there at it, the least significant bit D17 (0 or 1) is stored in latch 167 as a flag.
[0053] Next, program memory 180 is advanced to the n+2nd street, the 16th street of the address memory 120 by assignment of a pointer 121 is specified, and it is the address information (in this example). Read-out and a pointer 124 are minded for address data (100) and bit-position data (14). The 100th data ('D1 xx....x' interleaved last time is stored) of data memory 100 Read-out, In the logical operation machine 160, an AND is carried out, last time similarly, when a bit D17 is 0, with '1011....1', when a bit D17 is 1, an OR is carried out to '0100....0', and the result is stored in 170 registerr0. Thus, the data stored in 170 registerr0 serves as 'D1 1 xx...x', when a bit D17 is 0 and 'D1 0 xx...x' and a bit D17 are 1 (of course, this value of D1 is 0 or 1). A program can be hereafter advanced like the above-mentioned, a bit can be added, and an interleave can be performed.
[0054] The above procedure explained how interleave processing would be performed to the data D1-DNt before interleave processing. Next, the size of the storage region of the various memory needed in that case is explained. That is, since it becomes the repeat of the six above-mentioned step about program memory 180, about Nt bit and the address memory 120, it becomes [data memory / 100 / a 6xnickel bit (nickel is the number of bits of one word of program memory 180 and nickel is a positive integer), and] a NtxNa bit (Na is the number of bits of one word of the address memory 120, and Na is a positive integer).
[0055] For example, when Mr. field combination of the above-mentioned conventional technology, Nt=230, Nd=16, nickel=32, and Na=12, program memory 180 is needed nickelx6=32x6=192 bit, and Nt=230 bit, and NtxNa=320x 12= 2760 bits of address memory 120 are needed for data memory 100. Therefore, by all memory, 3182 bits is needed. It became clear that considerable saving of this invention was able to be carried out from 19320 bits needed with the above-mentioned conventional technology so that this number of bits might show.
[0056] Next, with reference to drawing 2 , and 5, 6, 8 and 11, it explains per [by the 2nd example of this invention] interleave equipment. The block diagram showing the configuration of interleave equipment according [drawing 2] to the 2nd example of this invention, the block diagram showing the data storage condition of the address memory which uses drawing 8 for interleave processing of **** 2 example, and drawing 11 are drawings showing the example of the program used for interleave processing of **** 2 example stored in program memory. In addition, about drawing 5 and drawing 6 , it is the same as that of the thing of the 1st above-mentioned example.
[0057] In drawing 2 , 100-188 are the same as what was shown in drawing 1 . the relative address information (the portion which changes relatively [the storing first-move ground] —) used in case 220 writes the data after interleave processing in data memory 100 Namely, the address memory holding the below-mentioned delta value and the below-mentioned bit-position information 123 from a pointer 240, The pointer with which 221 gives an access address to the address memory 220, the relative address information which outputs 222 from the address memory 220, The pointer holding the reference value or criteria address which consists of a portion common to each address with which 240 writes the data after interleave processing in data memory 100, 242 is an adder which outputs the address information used in case the relative address information 222 and the reference value from a pointer 240 are added and the data after interleave processing is written in data memory 100 to a pointer 124.
[0058] Hereafter, in the interleave equipment of this example constituted as mentioned above, the actuation in the case of performing an interleave is explained. As the interleave processing actuation shown below shall rearrange the data before the interleave processing stored in data memory 100 in the condition which showed in drawing 5 as shown in drawing 6 , and it shows it to drawing 5 The data D1 before interleave processing thru/or DNt make the 0th street a head address, and it is stored in order from the least significant bit to the most significant bit, and data memory 100 and a register 170 are assumed to be what is performed under the conditions of consisting of 16 bits (Nd=16).
[0059] the address memory 220 — the data bits D1-DNt before interleave processing — (it being hereafter called a relative

address) and its bit position of the portion which is alike, respectively and changes relatively [the address which is a storing place to the data memory 100 of the data after the receiving interleave processing] are relative -- it is stored sequentially from the 0th street. By this example, as shown in drawing 8 , it is stored in format as address data (0, 2, 1, ...) and the bit-position data (15, 1, 7, ...) corresponding to it show there. Moreover, 0 is beforehand stored in the pointer 221 and the reference value (for example, the first address and this example 100th street) used as the criteria at the time of writing the data after interleave processing in data memory 100 is stored in the pointer 240.

[0060] The program shown in drawing 11 is stored in program memory 180, and it is the following points that the actuation differs from what was shown in drawing 10 . Namely, although data was stored in read-out in the n-th street [n+4th] from the address of the address memory 120 specified with the pointer 121 in the 1st example shown in drawing 10 and the address data 122 was stored in the pointer 124 The data as actuation replaced with it from the address specified with the pointer 221 of the address memory 220 in this example Read-out, The relative address information 222 in it and the reference value (this example 100) of a pointer 240 are added in an adder 242, and it differs in that the result was stored in the pointer 124.

[0061] Address information which should be held by the address memory 220 can be made into the relative value which deducted the reference value from a pointer 240 by this, and the storage region of the address memory 220 can be made still smaller.

[0062] In addition, in the 1st and 2nd examples, although the data before interleave processing was arranged sequentially from the least significant bit, this is good even from the most significant bit. At this time, 1 bit shift of the shifter 165 will be carried out to a high order, and latch's 167 flag will hold the most significant bit of the output of a shifter 165. Moreover, the array of the data storage after interleave processing is arbitrary, and a good thing cannot be overemphasized.

[0063]

[Effect of the Invention] Constitute this invention, as explained above, and a storage means to memorize the bit-position information which specifies the location of the bit after interleave processing of the address data to the data storage region after interleave processing and its data especially is established. By having enabled it to carry out arbitration assignment of the address and the bit position of data after interleave processing separately using the address data and bit-position information While memorizing the bit before interleave processing throughout the data before interleave processing By having made possible addressing of the data before this interleave processing repeatedly [of the same program], since it is good at the program memory of a small storage region (small circuit scale), and use of small data memory, The memory areas which should be used for an interleave are sharply reducible. Furthermore, this invention can reduce the storage areas of a relative address further by constituting, as explained above, constituting so that a reference value common to all addresses and a changed part (relative address) of each address peculiar to each address may be added and the address data to the data storage region after interleave processing may be generated especially, and having memorized only the relative address in address memory.

[Translation done.]

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]
[Drawing 1] The block diagram of the interleave equipment by the 1st example of this invention
[Drawing 2] The block diagram of the interleave equipment by the 2nd example of this invention
[Drawing 3] The block diagram of conventional interleave equipment
[Drawing 4] Drawing showing the storing condition of the data before the interleave processing used with the conventional technology
[Drawing 5] Drawing showing the storing condition of the data before usable interleave processing by this invention
[Drawing 6] Drawing showing the storing condition of the data after interleave processing
[Drawing 7] Drawing showing the storing condition of the data of the address memory used in the 1st example of this invention
[Drawing 8] Drawing showing the storing condition of the data of the address memory used in the 2nd example of this invention
[Drawing 9] Drawing showing the example of the conventional program
[Drawing 10] Drawing showing the example of the program used in the 1st example of this invention
[Drawing 11] Drawing showing the example of the program used in the 2nd example of this invention
[Description of Notations]
100 Data Memory
110,121,124 Pointer
120 Address Memory
170 Register
122 Address Data
123 Bit-Position Information
130 Control Unit
132 Data
134,184,188 Control signal
150 Multiplexer
160 Logical Operation Machine
165 Shifter
167 Latch
180 Program Memory
182 Decoder
186 Control Section
220 Address Memory
221 Pointer
222 Relative Address Information
240 Pointer
242 Adder
300 Data Memory
324 Pointer
340 Shifter
342 Register
360 Logical Operation Machine
370 Register
380 Program Memory
382 Decoder

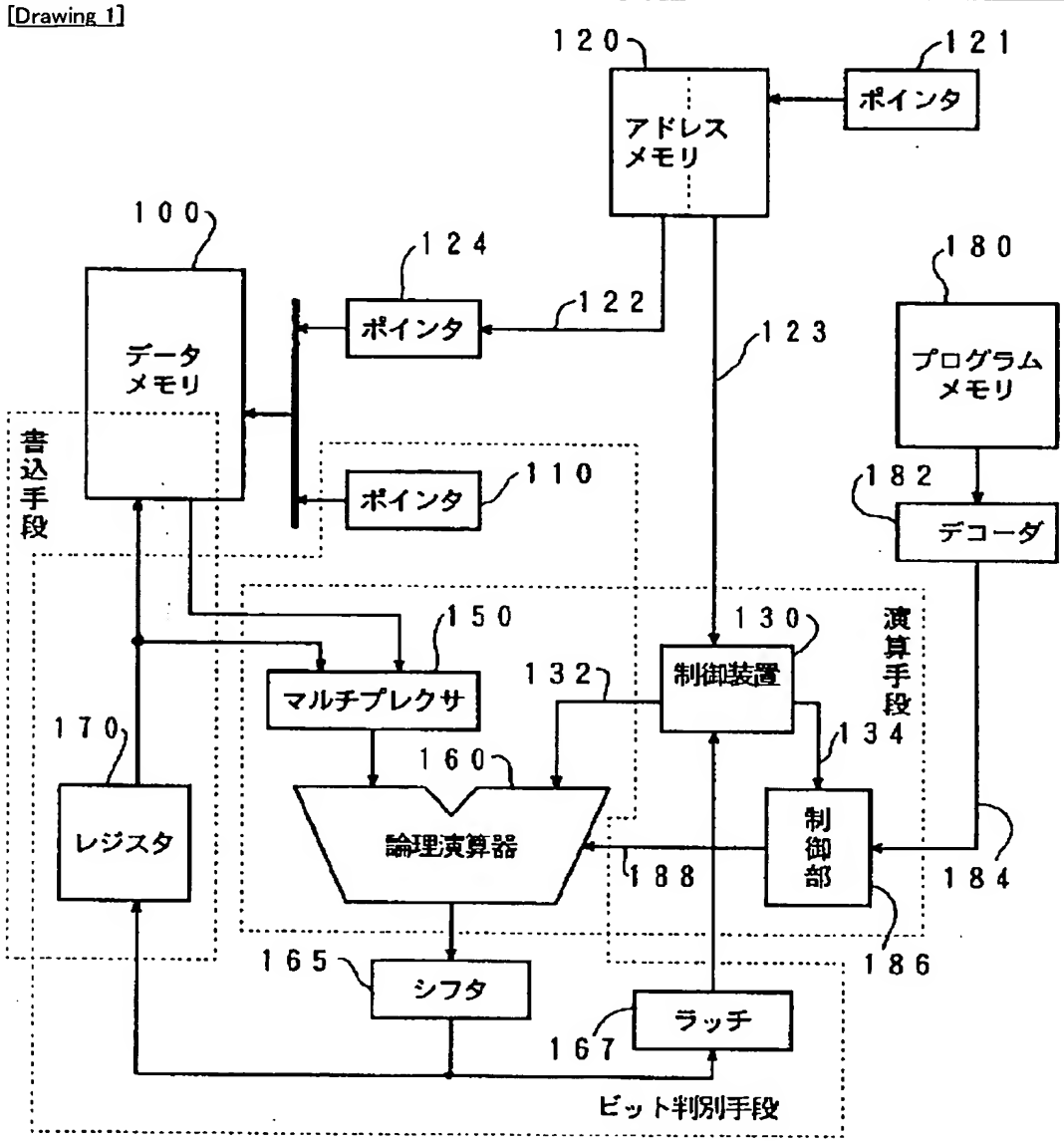
[Translation done.]

* NOTICES *

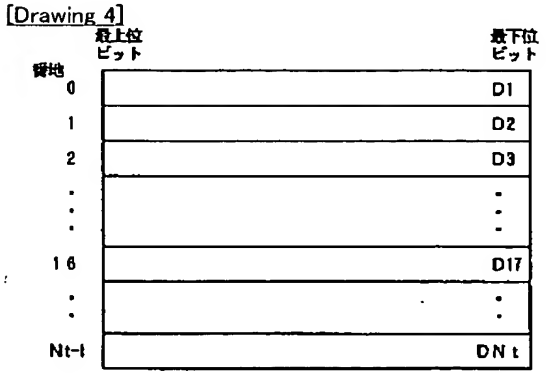
1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DRAWINGS

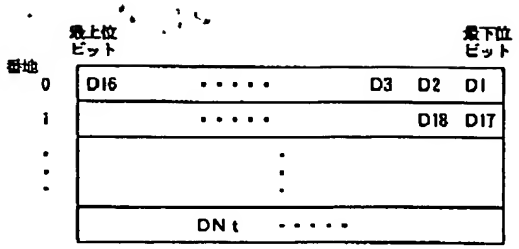
[Drawing 1]



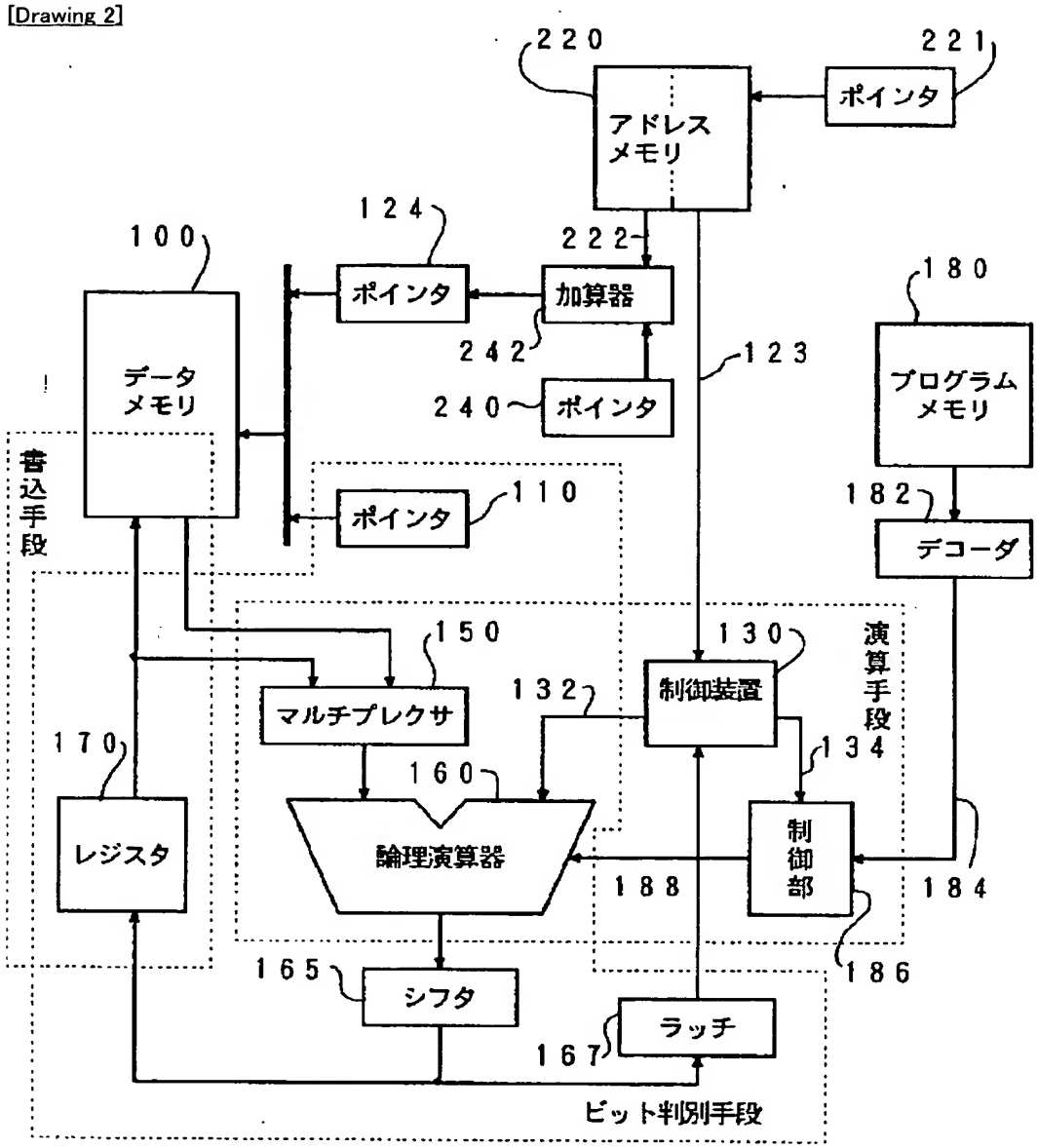
[Drawing 4]



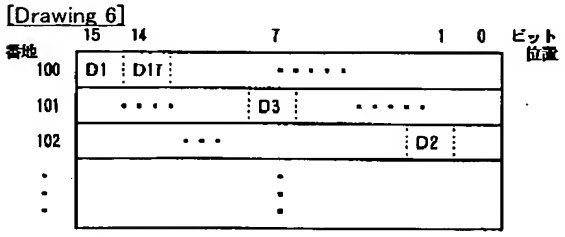
[Drawing 5]



[Drawing 2]



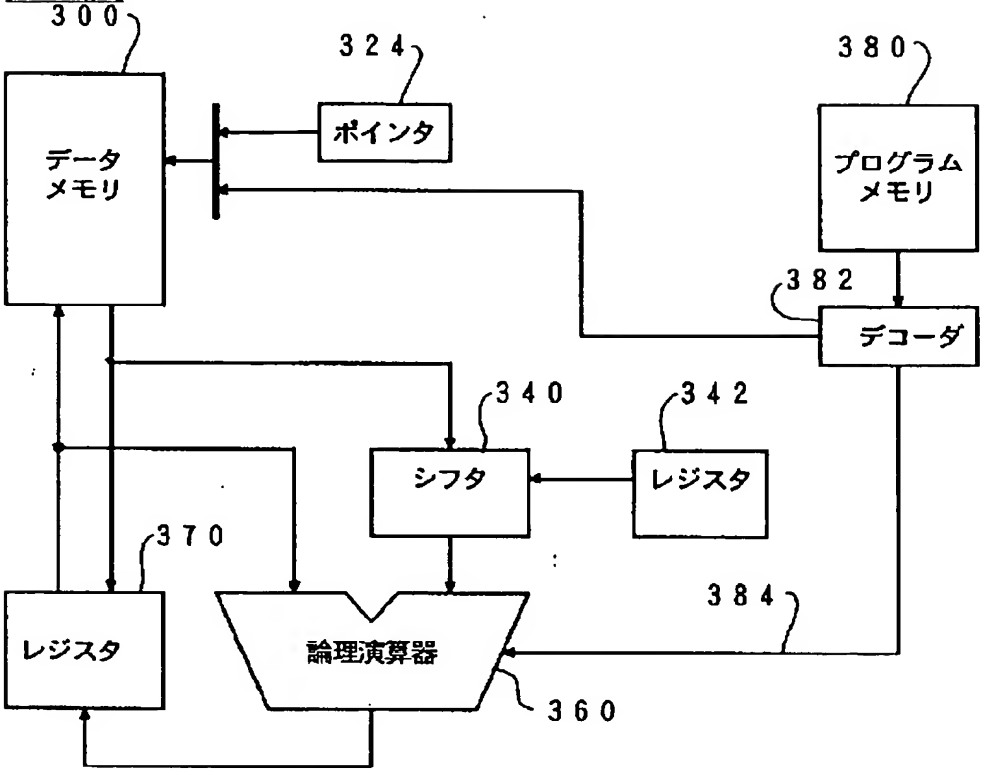
[Drawing 6]



[Drawing 7]

番地	番地データ	ビット位置データ
0	100	15
1	102	1
2	101	7
・	・	・
・	・	・
・	・	・
16	100	14
・	・	・

[Drawing 3]



[Drawing 8]

番地	番地データ	ビット位置データ
0	0	15
1	2	1
2	1	7
・	・	・
・	・	・
・	・	・
・	・	・

[Drawing 9]

番地	処理手順
n	レジスタ３７０ ← データメモリ３００（ポインタ３２４の番地）
n＋1	レジスタ３４２ ← シフトビット数
n＋2	レジスタ３７０ ← レジスタ３７０と（データメモリ３００（0番地）をシフト）の論理和
n＋3	レジスタ３４２ ← シフトビット数
n＋4	レジスタ３７０ ← レジスタ３７０と（データメモリ３００（１７番地）をシフト）の論理和
・	・
・	・
・	

：	：
n＋m	データメモリ３００（ポインタ３２４の番地） ←― レジスタ３７０ ポインタ３２４ ←― ポインタ３２４＋１

[Drawing 10]

番地	処理手順
n	ｒ１ ←― データメモリ１００（ポインタ１１０の番地） ポインタ１２４ ←― 番地データ１２２ ポインタ１１０ ←― ポインタ１１０＋１
n＋１	繰返し回数（Ｎｄ－１）の設定
n＋２	インターリーブ格納命令 ・ｒ０ ←― データメモリ１００（ポインタ１２４の番地）の特定ビットのラッチ１６７にフラグをセット ・レジスタ１２１ ←― レジスタ１２１＋１
n＋３	データメモリ１００（ポインタ１２４の番地） ←ｒ０
n＋４	ｒ１ ←― ｒ１を下位側に１ビットシフト ポインタ１２４ ←― 番地データ１２２ ｎ＋２番地に戻る
n＋５	ｎ番地に戻る

[Drawing 11]

番地	処理手順
n	ｒ１ ←― データメモリ１００（ポインタ１１０の番地） ポインタ１２４ ←― 相対的番地データ２２２＋ポインタ２４０ ポインタ１１０ ←― ポインタ１１０＋１
n＋１	繰返し回数（Ｎｄ－ｉ）の設定
n＋２	インターリーブ格納命令 ・ｒ０ ←― データメモリ１００（ポインタ１２４の番地）の特定ビットのラッチ１６７にフラグをセット ・レジスタ１２１ ←― レジスタ１２１＋１
n＋３	データメモリ１００（ポインタ１２４の番地） ←ｒ０
n＋４	ｒ１ ←― ｒ１を下位側に１ビットシフト ポインタ１２４ ←― 相対的番地データ２２２＋ポインタ２４０ ｎ＋２番地に戻る
n＋５	ｎ番地に戻る

[Translation done.]